VŠB – Technická univerzita Ostrava

Fakulta elektrotechniky a informatiky

Katedra aplikované matematiky

Vytvoření interaktivních programů pro výuku lineární algebry

Interactive programs for the teaching of linear algebra

2019 Wojciech Raczkowski

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra aplikované matematiky

Zadání bakalářské práce

Student: **Wojciech Raczkowski**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 1103R031 Výpočetní matematika

Téma: Vytvoření interaktivních programů pro výuku lineární algebry
Interactive programs for the teaching of linear algebra

Jazyk vypracování: čeština

Zásady pro vypracování:

Všichni studenti Fakulty elektrotechniky a informatiky na začátku svého bakalářského studia absolvují kurz lineární algebry. Lineární algebra je důležitá a zajímavá disciplína, která má řadu zajímavých aplikací. Například uveďme analýzu elektrických obvodů pomocí metody smyčkových proudů nebo uzlových napětí, analýzu dopravních toků v městech, výpočet rozložení tepla v tělese či šifrování textu.

Pro motivaci studia lineární algebry napomáhá, pokud si studenti mohou aplikace přímo vyzkoušet. Úkolem práce je seznámit se s vybranými základními aplikacemi lineární algebry a připravit interaktivní programy s těmito aplikacemi pro výuku kurzů lineární algebry.

Práce bude mít tyto části:
- Seznámení se se základními pojmy lineární algebry
- Seznámení se s vybranými aplikacemi lineární algebry (např. analýza elektrického obvodu, analýza dopravního toku, modelování rozložení tepla, šifrování)
- Výběr software pro tvorbu interaktivních programů
- Tvorba interaktivních programů s vybranými aplikacemi lineární algebry

Seznam doporučené odborné literatury:

Z. Dostál, V. Vondrák: Lineární algebra. Text vytvořený při realizaci projektu Matematika pro inženýry 21. století, Vysoká škola báňská - Technická univerzita Ostrava (2012).

M. Hussein: Linear Algebra and Engineering Applications. Učební text United Arab Emirates University (2009).

Webová stránka University of California, Davis - Applications of Linear Algebra (https://www.math.ucdavis.edu/~daddel/linear_algebra_appl/Applications/applications.html)

Dále dle pokynů vedoucího bakalářské práce.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **doc. Ing. Petr Beremlijski, Ph.D.**

Datum zadání: 01.09.2018

Datum odevzdání: 12.07.2019

prof. RNDr. Jiří Bouchala, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 12. července 2019 .....

Abstrakt

Tato bakalářská práce je zaměřená na seznámení se základními pojmy lineární algebry a jejich využití pro vytvoření interaktivních programů pro výuku lineární algebry. Uvedeme si několik vybraných metod a operací, které využijeme ve vytvořených programech. K vytvoření programů použijeme software Matlab a také jeho grafické prostředí GUIDE.

Klíčová slova: matice, inverzní matice, Gaussova eliminační metoda, lineární zobrazení, Matlab, interaktivní výukový program, lineární algebra

Abstract

This bachelor thesis is focused on the basic concepts of linear algebra and their use for designing of interactive programs for the teaching of linear algebra. We will present some selected methods and operations that will be used in created programs. For designing of programs we will use Matlab and its graphical interface called GUIDE.

Key Words: matrix, inverse matrix, Gaussian elimination method, linear mapping, Matlab, interactive educational program, linear algebra

Obsah

Seznam obrázků a tabulek

Seznam obrázků

Seznam tabulek

1 Úvod

V této bakalářské práci se seznámíme se základními pojmy lineární algebry a následným využitím těchto znalosti k vytvoření 3 aplikací – šifrování, dopravní tok a rozložení teploty.

Aplikace v této práci jsou inspirovány a podrobněji popsány v dokumentu [3].

Aplikace šifrování bude sloužit k zašifrování libovolného textu pomocí rozdělení textu do menších sad a následném násobení stanovenou maticí. K dešifrování využijeme vlastnosti inverzní matice, kterou si také představíme. Ukážeme si také vztah mezi šifrováním a lineárním zobrazením.

Aplikace dopravní tok nám poslouží k simulaci dopravy v systému ulic s jednou uzavřenou ulicí. Aplikace vypočte, která křižovatka bude nejvíce vytížená, a pomocí vypočítaných dat můžeme optimalizovat světelnou signalizaci. Řešení tohoto problému povede k vytvoření soustavy lineárních rovnic.

V aplikaci rozložení teploty si ukážeme čtvercovou desku, na kterou působí z každé strany různé teploty a my se pokusíme vypočítat rozložení teploty uvnitř desky. K řešení tohoto problému nahradíme funkci rozložení teploty na desce rozložením teploty pouze v bodech na rovnoměrné mřížce. To povede k vytvoření soustavy lineárních rovnic a následně k výpočtu pomocí Gaussovy eliminace.

Kromě popisu samotných aplikací uvedeme nakonec i popis jejich implementací v software Matlab.

2 Základní pojmy a operace

V této kapitole si představíme základní pojmy a operace, které budeme využívat v následujících kapitolách. Podrobněji se s těmito pojmy a důkazy uvedených tvrzení lze seznámit v dokumentu [1] a [2].

2.1 Matice a základní operace s maticemi

Definice 1:

Nechť jsou dány prvky $a_{11}, a_{12}, \dots, a_{mn}$ z dané množiny $\mathbb{R}$, jejíž prvky lze sčítat a násobit obdobně jako čísla. Prvky množiny $\mathbb{R}$ nazýváme také skaláry. Matice typu (m, n) (stručně $m \times n$ matice) je obdélníková tabulka

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix},$$

která má mn prvků a_{ij} uspořádaných do m řádků r_i^A a n sloupců s_j^A, takže

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{r_1^A} \\ \vdots \\ \boldsymbol{r_m^A} \end{pmatrix} = (\boldsymbol{s_1^A}, \quad \cdots, \quad \boldsymbol{s_n^A})$$

Stručně píšeme též $A = [a_{ij}]$.

Například a_{32} označuje prvek matice nacházející se v třetím řádku a druhém sloupci.

Příklad:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 7 \\ 4 & 9 & 2 \\ 6 & 1 & 5 \end{pmatrix}$$

Matice A je maticí o rozměru 4x3, protože obsahuje $m = 4$ řádků a $n = 3$ sloupců. V tomto případě prvek matice a_{32} je roven 9.

Po seznámení se s pojmem matice se podíváme na různé typy matic, se kterými se můžeme setkat.

Řádková matice – jako řádkovou matici nazýváme matici obsahující pouze jeden řádek a libovolný počet sloupců.

Sloupcová matice – obdobně jako řádková matice, tato matice má pouze jeden sloupec a jakýkoli počet řádků. Tuto matici lze chápat jako sloupcový vektor.

Čtvercová matice – jako čtvercovou matici nazýváme matici, která obsahuje stejný počet řádků a sloupců.

Transponovaná matice

Definice 2:

K matici A typu (m, n) *definujeme transponovanou matici* A^T *typu* (n, m) *předpisem:*

$$(\boldsymbol{A^T})_{\boldsymbol{ij}} = (\boldsymbol{A})_{\boldsymbol{ji}}.$$

V další části si představíme základní operace, které můžeme s maticemi provádět.

Sčítání a odečítaní matic

Definice 3:

Součet matic A a B stejného typu je matice $A + B$ *stejného typu jako A a B definovaná předpisem*

$$(A + B)_{ij} = (A)_{ij} + (B)_{ij}.$$

Z definice vyplývá, že sčítání dvou matic můžeme provádět pouze se dvěma maticemi stejných rozměrů $m \times n$. Součet dvou matic $A + B$ je opět matice stejného rozměru $m \times n$.

Příklad:

$$\begin{pmatrix} 1 & 3 \\ 1 & 0 \\ 1 & 2 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 7 & 5 \\ 2 & 1 \end{pmatrix} = \begin{pmatrix} 1+0 & 3+0 \\ 1+7 & 0+5 \\ 1+2 & 2+1 \end{pmatrix} = \begin{pmatrix} 1 & 3 \\ 8 & 5 \\ 3 & 3 \end{pmatrix}$$

Pro odečítání matic platí stejná pravidla jako pro sčítání a provádí se stejným způsobem.

Násobení matic

Definice 4:

Pokud A je matice o rozměrech $m \times n$ *a B je matice o rozměrech* $n \times p$*, jejich součin* $A \cdot B$ *je matice s rozměry* $m \times p$ *definovaná vztahem*

$$(A \cdot B)_{ij} = \sum_{r=1}^{n} a_{ir} b_{rj} = a_{i1} b_{1j} + a_{i2} b_{2j} + \cdots + a_{(in)} b_{nj},$$

pro všechny prvky (i, j) *výsledné matice.*

Příklad:

$$A \cdot B = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix} = \begin{pmatrix} (1 \cdot 1 + 2 \cdot 3 + 3 \cdot 5) & (1 \cdot 2 + 2 \cdot 4 + 3 \cdot 6) \\ (4 \cdot 1 + 5 \cdot 3 + 6 \cdot 5) & (4 \cdot 2 + 5 \cdot 4 + 6 \cdot 6) \end{pmatrix} = \begin{pmatrix} 22 & 28 \\ 49 & 64 \end{pmatrix}$$

Inverzní matice

Definice 5:

Nechť A je čtvercová matice. Jestliže existuje matice B tak, že

$$\boldsymbol{AB = BA = I},$$

pak se matice B nazývá inverzní maticí k matici A. Čtvercová matice, ke které existuje inverzní matice, se nazývá regulární. V opačném případě takovou matici nazýváme singulární.

Věta 1:

Ke každé regulární matici A existuje právě jedna inverzní matice. Důkaz viz [2].

Inverzní matici k matici A značíme A^{-1}.

Příklad: Matice B je inverzní k matici A:

$$A = \begin{pmatrix} 0 & 2 & -1 \\ 1 & -2 & 1 \\ -1 & -1 & 1 \end{pmatrix}$$

$$B = \begin{pmatrix} 1 & 1 & 0 \\ 2 & 1 & 1 \\ 3 & 2 & 2 \end{pmatrix}$$

Lze snadno ověřit, že $A \cdot B = I$:

$$\begin{pmatrix} 0 & 2 & -1 \\ 1 & -2 & 1 \\ -1 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 & 0 \\ 2 & 1 & 1 \\ 3 & 2 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

2.2 Soustavy lineárních rovnic a jejich řešení

Definice 6:

Soustavou m lineárních rovnic o n neznámých $x_1, \dots, x_n$ nazýváme množinu rovnic ve tvaru:

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1$$

$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2$$

$$\dots$$

$$a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_n,$$

kde $a_{11}, \dots, a_{mn}, b_1, \dots, b_m$ jsou reálná čísla. Tato soustava se nazývá systém m lineárních rovnic o n neznámých $x_1, \dots, x_n$ s koeficienty $a_{11}, \dots, a_{mn}$.

Možnosti řešení soustavy lineárních rovnic

1. Řešení pomocí Gaussovy eliminační metody

Při řešení soustavy pomocí Gaussovy eliminační metody nejdříve redukujeme matici na schodový tvar a tento krok nazýváme dopřednou redukcí. Následné řešení soustavy se schodovou maticí nazýváme zpětnou substitucí.

Ekvivalentní úpravy

Ekvivalentní úpravy slouží k řešení soustavy lineárních rovnic. Myšlenka spočívá v nahrazení soustavy jinou soustavou se stejným řešením, ale které můžeme jednoduše spočítat. Například v případě dvou rovnic o dvou neznámých můžeme pomocí ekvivalentních úprav získat soustavu obsahující rovnici s jednou neznámou, kterou už můžeme vyřešit nezávisle na druhé rovnici.

Ekvivalentními úpravami nazýváme:

Vzájemná záměna libovolných dvou řádků soustavy.

Vynásobení obou stran libovolné rovnice soustavy stejným nenulovým číslem.

Přičtení násobku některé rovnice soustavy k jiné rovnici.

Příklad:

$$-2x_1 + x_2 = 0$$

$$x_1 - 3x_2 = -10$$

Díky druhému a třetímu pravidlu ekvivalentních úprav můžeme druhou rovnici vynásobit dvěma a k ní přičíst rovnici první. Upravená rovnice bude mít tvar:

$$-2x_1 + x_2 = 0$$

$$-5x_2 = -20$$

Na druhém řádku nám zbyla pouze jedná neznámá, kterou můžeme snadno vypočítat a dosadit do první rovnice:

$$x_2 = 4$$

$$-2x_1 + 4 = 0$$

Výsledek tedy bude:

$$x_1 = 2$$

$$x_2 = 4$$

Maticový zápis soustavy

Maticový zápis slouží k ušetření práce při řešení soustavy lineárních rovnic takovým způsobem, že vynecháme opisování neznámých.

Soustavu lineárních rovnic můžeme zapsat pomocí matice

$$\left(\begin{array}{ccc|c} a_{11} & \cdots & a_{1n} & b_1 \\ \vdots & \ddots & \vdots & \vdots \\ a_{m1} & \cdots & a_{mn} & b_m \end{array}\right),$$

kterou nazýváme *rozšířená matice soustavy*. Levá strana tabulky se nazývá *matice soustavy*. Pravá strana tabulky se nazývá *pravá strana soustavy*.

Ekvivalentní úpravy prováděné na rozšířené matici soustavy se nazývají *elementární operace* nebo též *řádkové operace*. Pro elementární operace platí stejné pravidla jako pro ekvivalentní úpravy.

Vzájemná výměna dvou řádků matice.

Vynásobení řádku matice libovolným číslem různým od nuly.

Přičtení násobku jednoho řádku k jinému řádku.

Pro řešení rozšířené matice soustavy potřebujeme získat takzvaný *schodový tvar*.

Schodový tvar matice soustavy získáme elementárními operacemi tak, abychom dostali matici do tvaru, kde jsou první nenulové prvky řádků (vedoucí prvky) uspořádány jako schody klesající zleva doprava, přičemž žádné dva první nenulové prvky řádků nemohou být nad sebou a případné nulové řádky musí být dole.

Příklad schodového tvaru matice:

$$\begin{pmatrix} 1 & 2 \\ 0 & 5 \end{pmatrix}, \begin{pmatrix} 2 & 1 & 0 \\ 0 & 1 & 8 \\ 0 & 0 & 6 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 5 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{pmatrix}$$

Při úpravách matice soustavy do schodového tvaru můžeme dále podělit každý řádek vedoucím prvkem a pomocí ekvivalentních úprav upravíme matici dále tak, aby i nad vedoucím prvkem každého řádku byly nuly. Matice v tomto tvaru je potom v *normovaném schodovém tvaru.*

Při řešení soustavy lineárních rovnic se můžeme setkat se třemi možnými výsledky:

Soustava s jediným řešením.

Soustava, která má nekonečně mnoho řešení.

Soustava, která nemá řešení.

Příklad soustavy s jediným řešením:

$$2x_2 + 3x_3 = 2$$

$$x_2 + x_3 = 0$$

$$x_1 + x_3 = 4$$

Nejdříve si soustavu převedeme na rozšířenou matici soustavy:

$$\left(\begin{array}{ccc|c} 0 & 2 & 3 & 2 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 4 \end{array}\right)$$

Ekvivalentními úpravami se postupně dostaneme do schodového tvaru matice:

$$\left(\begin{array}{ccc|c} 0 & 2 & 3 & 2 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 4 \end{array}\right)^{r3}_{r1} \rightarrow \left(\begin{array}{ccc|c} 1 & 0 & 1 & 4 \\ 0 & 1 & 1 & 0 \\ 0 & 2 & 3 & 2 \end{array}\right)_{r3-2r_2} \rightarrow \left(\begin{array}{ccc|c} 1 & 0 & 1 & 4 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 2 \end{array}\right)$$

Schodovou matici, kterou jsme získali si můžeme převést zpět na soustavu lineárních rovnic:

$$x_1 + x_3 = 4$$

$$x_2 + x_3 = 0$$

$$x_3 = 2$$

Jednoduchým vyřešením soustavy získáme výsledek:

$$x_1 = 2$$

$$x_2 = -2$$

$$x_3 = 2$$

Příklad soustavy s nekonečně mnoho řešeními:

$$x_1 + x_2 + x_3 = 1$$

$$-x_2 - 2x_3 = 0$$

Po upravení soustavy dostaneme:

$$x_1 = 1 + x_3$$

$$x_2 = -2x_3$$

Soustava má tedy nekonečně mnoho řešení, protože nemáme jednoznačný výsledek pro x_3. Výsledek můžeme zapsat také pomocí parametru p, ve tvaru $x_3 = p$. V tom případě výsledek bude vypadat následovně:

$$x_1 = 1 + p$$

$$x_2 = -2p$$

$$x_3 = p \, \epsilon \, \mathbb{R}$$

Příklad soustavy, která nemá řešení:

$$2x_1 + x_2 = 2$$

$$x_1 + 2x_2 - x_3 = 1$$

$$4x_1 + 5x_2 - 2x_3 = -1$$

Pomocí ekvivalentních úprav rozšířené matice soustavy dostaneme

$$\left(\begin{array}{ccc|c} 2 & 1 & 0 & 2 \\ 1 & 2 & -1 & 1 \\ 4 & 5 & -2 & -1 \end{array}\right)_{r_3 - 2r_1} \quad 2r_2 - r_1 \rightarrow \left(\begin{array}{ccc|c} 2 & 1 & 0 & 2 \\ 0 & 3 & -2 & 0 \\ 0 & 3 & -2 & -5 \end{array}\right)_{r_3 - r_2} \rightarrow \left(\begin{array}{ccc|c} 2 & 1 & 0 & 2 \\ 0 & 3 & -2 & 0 \\ 0 & 0 & 0 & -5 \end{array}\right)$$

Z posledního řádku rozšířené matice soustavy dostaneme rovnici:

$$0x_1 + 0x_2 + 0x_3 = -5$$

Tato soustava tedy nemá žádné řešení.

Výpočetní náročnost řešení pomocí Gaussovy eliminační metody je cca $\frac{1}{3}(n^3)$ operací násobení pro soustavy n lineárních rovnic o n neznámých.

2. Řešení pomocí inverzní matice

Soustavu lineárních rovnic můžeme vyřešit pomocí inverzní matice tak, že eliminujeme vektor neznámých z maticového zápisu soustavy a dosadíme:

$$A \cdot x = b$$

$$A^{-1} \cdot A \cdot x = A^{-1} \cdot b$$

$$x = A^{-1} \cdot b$$

V tomto případě stačí, když nalezneme inverzní matici k matici soustavy A. Inverzní matici nalezneme pomocí elementárních řádkových úprav. Matici A transformujeme na jednotkovou matici a vytvořená transformační matice, která vznikne uplatněním stejných elementárních řádkových úprav na jednotkovou matici, je inverzní matici.

$$(A|I) \approx (I|A^{-1})$$

Výpočetní náročnost nalezení inverzní matice je cca n^3 operací násobení pro matici řádu n a náročnost řešení (násobení inverzní maticí) je cca n^2 operací násobení.

Výše uvedený postup lze použít pouze, pokud je matice A regulární a soustava má právě jedno řešení.

Nalezení inverzní matice

Věta 2:

Nechť A je čtvercová matice řádu n. Pak rovnice

$$\boldsymbol{AX = I}$$

má jediné řešení X právě tehdy, když A je regulární. V tom případě platí $X = A^{-1}$. Důkaz viz [2].

Pokud A je regulární a $AX = I$, má tato rovnice jediné řešení a rozšířenou matici $[A|I]$ můžeme pomocí ekvivalentních řádkových úprav dostat do tvaru $[I|B]$, kde B je inverzní matice k matici A.

Příklad:

Nejdříve si zvolíme matici, ke které chceme najít matici inverzní.

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

V dalším kroku si vytvoříme jednotkovou matici se stejnými rozměry.

$$\left(\begin{array}{cc|cc} 1 & 2 & 1 & 0 \\ 3 & 4 & 0 & 1 \end{array}\right)$$

Následně začínáme upravovat levou stranu matice takto, abychom z ní dostali jednotkovou matici. Stejné úpravy provádíme i na pravé matici. V prvním kroku chceme dosáhnout nuly v levém dolním rohu matice A. Toho dosáhneme přičtením -3 násobku prvního řádku ke druhému.

$$\left(\begin{array}{cc|cc} 1 & 2 & 1 & 0 \\ 0 & -2 & -3 & 1 \end{array}\right)$$

V dalším kroku potřebujeme dostat nulu na pozici a_{12}, kde v tomto okamžiku máme 2. K tomuto nám postačí, když k prvnímu řádku přičteme řádek druhý.

$$\left(\begin{array}{cc|cc} 1 & 0 & -2 & 1 \\ 0 & -2 & -3 & 1 \end{array}\right)$$

Jediné, co nám zbylo je -2 ve druhém řádku matice A. Abychom z ní udělali jedničku, stačí celý řádek vydělit právě -2.

$$\left(\begin{array}{cc|cc} 1 & 0 & -2 & 1 \\ 0 & 1 & \frac{3}{2} & -\frac{1}{2} \end{array}\right)$$

Na levé straně matice jsme dosáhli jednotkové matice, co znamená, že matice po pravé straně je matice inverzní k matici A, respektive A^{-1}.

Pro kontrolu můžeme obě matice vynásobit, čímž bychom měli dostat znovu matici jednotkovou.

$$\begin{pmatrix} -2 & 1 \\ \frac{3}{2} & -\frac{1}{2} \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

2.3 Lineární zobrazení

Definice 7:

Nechť U, V jsou vektorové prostory. Zobrazení $A: U \to V$ se nazývá lineární zobrazení (operátor), jestliže pro každé dva vektory $u, v \in U$ skalár $\alpha \in \mathbb{R}$ platí:

1. $A(u+v) = A(u) + A(v)$,
2. $A(\alpha u) = \alpha A(u)$.

Lineární zobrazení $U \to U$ se často nazývá lineární transformace. Množinu všech lineárních zobrazení vektorového prostoru U do vektorového prostoru V budeme značit $L(U,V)$. Místo $L(U,U)$ budeme psát $L(U)$.

Matice lineárního zobrazení

Věta 3:

Nechť $A: \mathbb{R}^{m,1} \to \mathbb{R}^{n,1}$ je libovolné lineární zobrazení. Pak existuje matice A typu (n,m) tak, že pro libovolné $x \in \mathbb{R}^{m,1}$ platí:

$$A(x) = Ax.$$ Důkaz viz [1] a [2].

Dále předpokládejme, že vektorové prostory U,V jsou konečné dimenze s bázemi

$$\varepsilon = (e_1, \dots, e_m) \text{ a } F = (f_1, \dots, f_n).$$

Definice 8:

Nechť $A: U \to V$ je lineární zobrazení. Pak můžeme vektory $A(e_1), \dots, A(e_m)$ vyjádřit jako lineární kombinace vektorů $f_1, \dots, f_n$ ve tvaru:

$$\begin{array}{ccccc} A(e_1) = & a_{11}f_1 & + \cdots + & a_{n1}f_1 \\ \vdots & \vdots & \cdots & \vdots \\ A(e_m) = & a_{1m}f_1 & + \cdots + & a_{nm}f_n \end{array}$$

Matici $[A]_{\varepsilon,F} = [a_{ij}] = [[A(e_1)]_F, \dots, [A(e_m)]_F]$ nazýváme maticí lineárního zobrazení A vzhledem k bázím (ε, F). Jestliže $U = V$ a $\varepsilon = F$, pak budeme mluvit o matici lineární transformace vzhledem k bázi ε a budeme jí značit $[A]_\varepsilon$.

Věta 4:

Nechť $A: U \to V$ je lineární zobrazení, ε je báze U a F je báze V. Pak pro libovolné $x \in U$ platí

$$[A(x)]_F = [A]_{\varepsilon,F}[x]_\varepsilon.$$ Důkaz viz [1] a [2].

Příklad:

Máme dáno zobrazení $A: \mathbb{R}^3 \to \mathbb{R}^2$ definované předpisem:

$$A([x_1, x_2, x_3]) = [x_1 - x_2, x_2 + x_3]$$

Rozhodněme, zda je zobrazení lineární.

V definici 7 máme 2 podmínky, které lineární zobrazení musí splňovat:

1. $\forall u, v \in \mathbb{R}^3: A(u+v) = A(u) + A(v)$

Zvolíme $u = [u_1, u_2, u_3], v = [v_1, v_2, v_3]$.

$A(u+v) = A([u_1 + v_1, u_2 + v_2, u_3 + v_3]) = [(u_1 + v_1) - (u_2 + v_2), (u_2 + v_2) + (u_3 + v_3)] = [u_1 - u_2 + v_1 - v_2, u_2 + u_3 + v_2 + v_3] = [u_1 - u_2, u_2 + u_3] + [v_1 - v_2, v_2 + v_3] = A(u) + A(v)$.

2. $\forall \alpha \ni \mathbb{R}\, \forall u \in \mathbb{R}^3 : A(\alpha u) = \alpha A(u)$.

$A(\alpha u) = A([\alpha u_1, \alpha u_2, \alpha u_3]) = [\alpha u_1 - \alpha u_2, \alpha u_2 + \alpha u_3] = [\alpha(u_1 - u_2), \alpha(u_2 + u_3)] = \alpha[u_1 - u_2, u_2 + u_3] = \alpha A(u)$.

Z 1. a 2. vyplývá, že zobrazení A je lineární.

3 Šifrování

V této kapitole si představíme způsob šifrování pomocí rozdělení textu do menších části a následnému využití lineární algebry.

3.1 Úvod

Dějiny kryptografie sahají až několik tisíc let zpátky. Kryptografií můžeme rozdělit na dvě části:

- klasická kryptografie – k šifrování stačila tužka a papír a jednoduché pomůcky, tato kryptografie převažovala až do první poloviny 20. století, než se začaly produkovat stroje, které umožňovaly více sofistikované šifrování.

- moderní kryptografie – s nástupem moderních technologií se objevily nové možnosti šifrování, které umožňovaly daleko složitější postup, jako například Enigma [6], což byl přenosný šifrovací stroj vynalezen ve dvacátých letech 20. století. Nejdříve se používal pro šifrování civilních zpráv a následně ho začaly používat i armády jako třeba Německo za druhé světové války.

V dnešní době se zvlášť produkované přístroje pro šifrování běžně nepoužívají, protože k tomu stačí běžné počítače.

Více na téma kryptografie a dějin kryptografie se můžeme dočíst v [5].

3.2 Šifrování

V této kapitole si představíme postup, který umožňuje uživateli zašifrovat a dešifrovat vybranou zprávu či text.

Vybral jsem si šifrování pomocí rozdělení čistého textu do sad o 3 písmenech a jejich následné nahrazení příslušnými číslicemi. Tento způsob šifrování je spíše učebnicového charakteru a běžně se nepoužívá, jelikož pro opravdové šifrování se používají daleko pokročilejší metody.

Abychom mohli pokračovat se šifrováním, musíme si nejdříve vybrat invertibilní matici, ke které si nalezneme matici inverzní, kterou budeme následně potřebovat k dešifrování zašifrované zprávy.

V tomto zadání jsme si vybrali tuto matici 3x3 a její inverzní matici:

$$A = \begin{pmatrix} 0 & 2 & -1 \\ 1 & -2 & 1 \\ -1 & -1 & 1 \end{pmatrix}$$

$$A^{-1} = \begin{pmatrix} 1 & 1 & 0 \\ 2 & 1 & 1 \\ 3 & 2 & 2 \end{pmatrix}$$

Aby tento program měl smysl a obě strany se mohly dorozumět, každá ze stran musí znát obě matice, jednu pro zašifrování a jednu pro dešifrování zprávy. Dále každá ze stran musí znát asociaci písmen a dalších znaků s číslicemi.

Nejdříve si přidružíme každé číslo pomocí tabulky 1 s šifrovacím znakem (první je mezera):

Tabulka 1: Přidružení čísel k písmenům.

" "	0	n	14	B	28	P	42	-	56	6	70
a	1	o	15	C	29	Q	43	*	57	7	71
b	2	p	16	D	30	R	44	=	58	8	72
c	3	q	17	E	31	S	45	/	59	9	73
d	4	r	18	F	32	T	46	!	60		
e	5	s	19	G	33	U	47	%	61		
f	6	t	20	H	34	V	48	(	62		
g	7	u	21	I	35	W	49	)	63		
h	8	v	22	J	36	X	50	0	64		
i	9	w	23	K	37	Y	51	1	65		
j	10	x	24	L	38	Z	52	2	66		
k	11	y	25	M	39	.	53	3	67		
l	12	z	26	N	40	,	54	4	68		
m	13	A	27	O	41	+	55	5	69		

Strana, která chce poslat zašifrovanou zprávu, tak musí nejdříve zaměnit jednotlivé znaky za řetěz číslic. Následně se tento řetěz rozdělí do vektorů o velikosti 3x1 a každý z těchto vektorů se vynásobí naší původní invertibilní matici, čímž se nám vytvoří zašifrované vektory 3x1, ze kterých vytvoříme konečný řetězec a zašifrována zpráva je hotová.

Ukažme si to na následujícím příkladu.

Řekněme, že chceme zašifrovat zprávu „ahoj karle“.

Nejdříve provedeme záměnu písmen na číslice:

a	h	o	j		k	a	r	l	e
1	8	15	10	0	11	1	18	12	5

Tento řetězec číslic následně naskládáme do vektorů o velikosti 3x1:

$$\begin{pmatrix}1\\8\\15\end{pmatrix},\begin{pmatrix}10\\0\\11\end{pmatrix},\begin{pmatrix}1\\18\\12\end{pmatrix},\begin{pmatrix}5\\\\\end{pmatrix}$$

Ihned si všimněme, že ne vždy je délka zprávy dělitelná třemi, čímž se nám komplikuje následné násobení s matici. Tomuhle můžeme předejít tím, že na konec řetězce vložíme potřebné množství nul abychom zaplnili poslední vektor.

Hotové vektory, které se pošlou k násobení pak vypadají následovně:

$$\begin{pmatrix}1\\8\\15\end{pmatrix},\begin{pmatrix}10\\0\\11\end{pmatrix},\begin{pmatrix}1\\18\\12\end{pmatrix},\begin{pmatrix}5\\0\\0\end{pmatrix}$$

Po vynásobení těchto vektorů maticí A dostaneme skupinu zašifrovaných vektorů:

$$A = \begin{pmatrix} 0 & 2 & -1 \\ 1 & -2 & 1 \\ -1 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 8 \\ 15 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 6 \end{pmatrix}$$

$$A = \begin{pmatrix} 0 & 2 & -1 \\ 1 & -2 & 1 \\ -1 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 10 \\ 0 \\ 11 \end{pmatrix} = \begin{pmatrix} -11 \\ 21 \\ 1 \end{pmatrix}$$

$$A = \begin{pmatrix} 0 & 2 & -1 \\ 1 & -2 & 1 \\ -1 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 18 \\ 12 \end{pmatrix} = \begin{pmatrix} 24 \\ -23 \\ -7 \end{pmatrix}$$

$$A = \begin{pmatrix} 0 & 2 & -1 \\ 1 & -2 & 1 \\ -1 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 5 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 5 \\ -5 \end{pmatrix}$$

Skupina zašifrovaných vektorů pak vypadá následovně:

$$\begin{pmatrix} 1 \\ 0 \\ 6 \end{pmatrix}, \begin{pmatrix} -11 \\ 21 \\ 1 \end{pmatrix}, \begin{pmatrix} 24 \\ -23 \\ -7 \end{pmatrix}, \begin{pmatrix} 0 \\ 5 \\ -5 \end{pmatrix}$$

Vektory rozložíme do řetězce a vznikne nám zakódovaná zpráva: 1 0 6 -11 21 1 24 -23 -7 0 5 -5.

Šifrování tedy provádíme pomocí násobení aritmetických vektorů matici, která je šifrovacím klíčem. Šifrování lze také chápat jako hledání obrazu v daném lineárním zobrazení, které je popsáno maticí – šifrovacím klíčem.

3.3 Dešifrování

Strana, která chce zprávu dešifrovat, za předpokladu, že zná inverzní matici a asociace znaků s čísly, tak musí nejdříve zašifrovanou zprávu rozdělit do vektorů 3x1, následně násobit každý vektor s inverzní matici původní matice a zaměnit čísla s písmeny.

Příklad:

Vezměme si řetězec z minulého příkladu a předpokládejme, že jsme dostali zašifrovanou zprávu:

1 0 6 -11 21 1 24 -23 -7 0 5 -5

Nejdříve si zprávu rozdělíme do sloupcových vektorů 3x1. Pokud byla zpráva správně zašifrována naším programem, tak počet členů je dělitelný třemi, tím pádem se nám naplní všechny vektory:

$$\begin{pmatrix}1\\0\\6\end{pmatrix},\begin{pmatrix}-11\\21\\1\end{pmatrix},\begin{pmatrix}24\\-23\\-7\end{pmatrix},\begin{pmatrix}0\\5\\-5\end{pmatrix}$$

Po vynásobení těchto vektorů maticí A^{-1} dostaneme výsledné vektory, ze kterých už můžeme udělat řetězec a zaměnit číslice na písmena.

$$A^{-1}=\begin{pmatrix}1&1&0\\2&1&1\\3&2&2\end{pmatrix}\cdot\begin{pmatrix}1\\0\\6\end{pmatrix}=\begin{pmatrix}1\\8\\15\end{pmatrix}$$

$$A^{-1}=\begin{pmatrix}1&1&0\\2&1&1\\3&2&2\end{pmatrix}\cdot\begin{pmatrix}-11\\21\\1\end{pmatrix}=\begin{pmatrix}10\\0\\11\end{pmatrix}$$

$$A^{-1}=\begin{pmatrix}1&1&0\\2&1&1\\3&2&2\end{pmatrix}\cdot\begin{pmatrix}24\\-23\\-7\end{pmatrix}=\begin{pmatrix}1\\18\\12\end{pmatrix}$$

$$A^{-1}=\begin{pmatrix}1&1&0\\2&1&1\\3&2&2\end{pmatrix}\cdot\begin{pmatrix}0\\5\\-5\end{pmatrix}=\begin{pmatrix}5\\0\\0\end{pmatrix}$$

Po vytažení číslic z vektoru dostaneme řetězec 1 8 15 10 0 11 1 18 12 5 0 0. Tento řetězec už můžeme jednoduše zaměnit na příslušné znaky z původní tabulky.

1	8	15	10	0	11	1	18	12	5	0	0
a	h	o	j		k	a	r	l	e		

Záměnou číslic za znaky jsme dostali náš původní text a ke konci dvě prázdná políčka, která se vytvořila z původního textu, protože délku textu jsme potřebovali mít dělitelnou třemi, abychom mohli naplnit všechny vektory.

Na šifrování se můžeme také podívat z hlediska lineárního zobrazení jako zobrazení mezi dvěma vektorovými prostory. Matice, ve které máme uložený klíč pro šifrování je vlastně maticí lineární transformace. Báze ε *a* F si zvolíme:

$$\varepsilon = F = ([1,0,0],[0,1,0],[0,0,1]).$$

Následně hledáme lineární zobrazení $x \to Ax$, kde $x \in \mathbb{R}^3, Ax \in \mathbb{R}^3$.

Maticí A máme definovanou klíčem pro šifrování:

$$A=\begin{pmatrix}0&2&-1\\1&-2&1\\-1&-1&1\end{pmatrix},$$

z ní si určíme skaláry $\propto_{11},\dots,\propto_{33}$ jako

$$\begin{array}{lll} \propto_{11}= 0 & \propto_{12}= 2 & \propto_{13}= -1 \\ \propto_{21}= 1 & \propto_{22}= -2 & \propto_{23}= 1 \\ \propto_{31}= -1 & \propto_{32}= -1 & \propto_{33}= 1. \end{array}$$

Předpis pro lineární zobrazení definujeme jako:

$$A([x_1, x_2, x_3]) = [\propto_{11} x_1 + \propto_{12} x_2 + \propto_{13} x_3, \propto_{21} x_1 + \propto_{22} x_2 + \propto_{23} x_3, \propto_{31} x_1 + \propto_{32} x_2 + \propto_{33} x_3]$$

Po dosazení skalárů $\propto_{11}$, ... $\propto_{33}$ dostaneme předpis pro lineární zobrazení:

$$A([x_1, x_2, x_3]) = [2x_2 - x_3, x_1 - 2x_2 + x_3, -x_1 - x_2 + x_3].$$

Vyzkoušejme si to na příkladu slova „vsb“. Podle tabulky 1 si postupně zaměníme písmena na číslice: v = 22 = x_1, s = 19 = x_2, b = 2 = x_3.

Následně dosadíme do předpisu pro lineární zobrazení:

$$A([x_1, x_2, x_3]) = [2 \cdot 19 - 2, 22 - 2 \cdot 19 + 2, -22 - 19 + 2] = [36, -14, -39].$$

Zašifrována zpráva tedy bude mít tvar: 36 -14 -39.

Tuto zašifrovanou zprávu také dostaneme, když vektor $\begin{pmatrix} 22 \\ 19 \\ 2 \end{pmatrix}$ vynásobíme šifrovací maticí

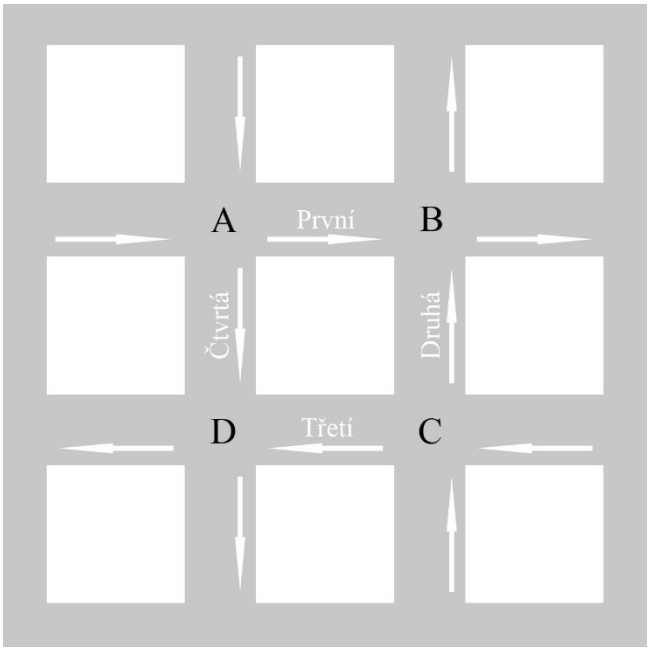
$A = \begin{pmatrix} 0 & 2 & -1 \\ 1 & -2 & 1 \\ -1 & -1 & 1 \end{pmatrix}$.

4 Dopravní tok

V této kapitole si představíme řešení problému dopravního toku pomocí soustavy lineárních rovnic.

4.1 Zadání

V dopravní špičce dochází k přetížení dopravy na křižovatkách ulic jako na obrázku níže.



Představme si, že město si přeje vylepšit plynulost dopravy a přizpůsobit světelnou signalizaci. V našem případě jsou všechny ulice jednosměrky příslušně označeny šipkami.

Nejdříve se z monitoringu nasbíraly data ohledně počtu přijíždějících a vyjíždějících aut na každé křižovatce za dobu jedné hodiny v odpolední špičce:

1. Křižovatka A:

Počet vozidel jedoucích po První ulici, které vjíždí na křižovatku A: 700.

Počet vozidel jedoucích po Čtvrté ulici, které vjíždí na křižovatku A: 300.

2. Křižovatka B:

Počet vozidel opouštějících křižovatku B po První ulici: 200.

Počet vozidel opouštějících křižovatku B po Druhé ulici: 900.

3. Křižovatka C:

Počet vozidel jedoucích po Třetí ulici, které vjíždí na křižovatku C: 400.

Počet vozidel jedoucích po druhé ulici, které vjíždí na křižovatku C: 300.

4. Křižovatka D:

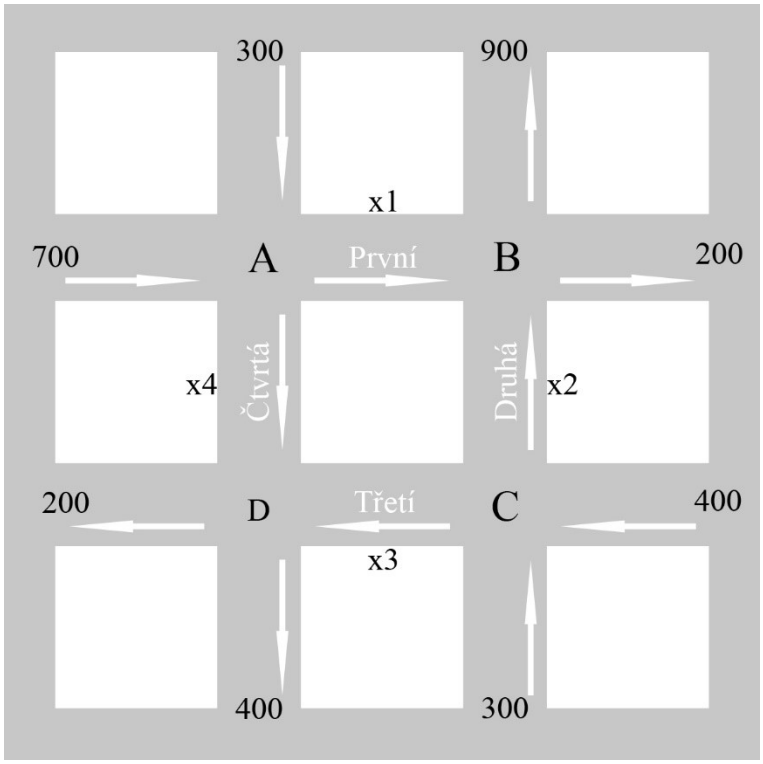
Počet vozidel opouštějících křižovatku D po Třetí ulici: 200.

Počet vozidel opouštějících křižovatku D po Čtvrté ulici: 400.

4.2 Postup řešení

Označení směrů:

Nejdříve si na obrázky vyznačíme naměřené hodnoty pro přehlednost a následně si označíme dopravní toky mezi jednotlivými křižovatkami.



x_1 – počet vozidel vyjíždějících z křižovatky A směrem ke křižovatce B.

x_2 – počet vozidel vyjíždějících z křižovatky C směrem ke křižovatce B.

x_3 – počet vozidel vyjíždějících z křižovatky C směrem ke křižovatce D.

x_4 – počet vozidel vyjíždějících z křižovatky A směrem ke křižovatce D.

Abychom tento problém mohli vyřešit, musíme předpokládat následující:

1. Každé vozidlo přijíždějící na křižovatku z něj rovněž odjede. V tomto případě se počet vozidel přijíždějících na křižovatku rovná počtu vozidel opouštějících křižovatku.

2. Všechny ulice jsou jednosměrky.

3. Všechny proměnné x_1, x_2, x_3 a x_4 jsou nezáporné, protože reprezentují počet vozidel projíždějících ulicí v předepsaném směru a ten nemůže být záporný.

Použitím prvního předpokladu dostaneme 4 rovnice o 4 neznámých:

Křižovatka A: $x_1 + x_4 = 700 + 300$

Křižovatka B: $x_1 + x_2 = 900 + 200$

Křižovatka C: $x_2 + x_3 = 400 + 300$

Křižovatka D: $x_3 + x_4 = 400 + 200$

Tyto čtyři rovnice vyřešíme pomocí Gaussovy eliminační metody:

$$x_1 + x_4 = 1000$$

$$x_1 + x_2 = 1100$$

$$x_2 + x_3 = 700$$

$$x_3 + x_4 = 600$$

Řešením této soustavy se dostaneme k výsledku:

$$x_1 = 1000 - x_4$$

$$x_2 = 100 + x^4$$

$$x_3 = 600 - x_4$$

$$x_4 \in \mathbb{R}$$

Soustava nemá jednoznačné řešení a to znamená, že nejsme schopní jednoznačně určit a naplánovat dopravu pro tyto ulice.

4.3 Aplikace

V předchozím příkladu jsme počítali se situací, kde všechny křižovatky a ulice jsou v provozu a v tomto případě jsme se nedostali k jednoznačnému výsledku. Pojďme se ale podívat na případ, když se město rozhodne, že je potřeba některou z ulic uzavřít kvůli rekonstrukci, či nějaké náhlé události. Uzavřeme třeba Třetí ulici, kterou představuje proměnná x_3. Z naměřených hodnot opět získáme podobnou soustavu jako v předchozím příkladu, jenže tentokrát hodnota x_3 bude nulová:

$$x_1 + x_4 = 1000$$

$$x_1 + x_2 = 1100$$

$$x_2 + x_3 = 700$$

$$x_3 + x_4 = 600$$

$$x_3 = 0$$

Vyřešením této soustavy se dostaneme k výsledku:

$$x_1 = 400$$

$$x_2 = 700$$

$$x_3 = 0$$

$$x_4 = 600$$

Z křižovatky A vyjíždí počet vozidel představující proměnné x_1 a x_4:

Z křižovatky A tedy vyjede 400 vozidel směrem ke křižovatce B a 600 vozidel směrem ke křižovatce D.

Na křižovatku B přijíždí počet vozidel představující proměnné x_1 a x_2:

Na křižovatku B tedy přijede 400 vozidel po První ulici směrem z křižovatky A a 700 vozidel po Druhé ulici směrem z křižovatky C.

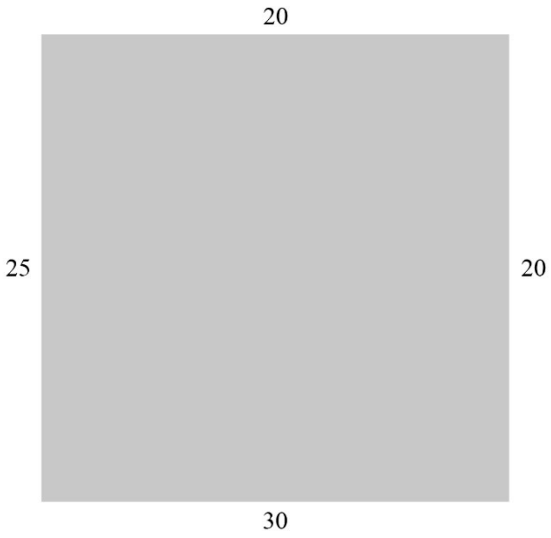
Obdobně z křižovatky C vyjede 700 vozidel směrem ke křižovatce B.

Na křižovatku D přijede 600 vozidel směrem z křižovatky A.

Dopravní tok ve městě popisujeme soustavou rovnic, kterou následně vyřešíme Gaussovou eliminační metodou.

5 Distribuce teploty

Představme si průřez čtvercové kovové desky. Každá hrana desky má stálou teplotu. Čísla kolem hran představují teplotu hran ve stupních Celsia. Nás zajímá rozdělení teplot uvnitř desky. Předpokládáme, že teploty na hranách desky jsou konstantní. Následující obrázek nám přiblíží situaci.



Uvažujeme úlohu vedení tepla na čtverci o straně délky L.

Laplaceova rovnice:

$$\begin{cases} -\Delta u = 0 \; pro \; \forall (x,y) \in \Omega = (0,L) \times (0,L) \\ u(x,y) = f_i \; pro \; \forall (x,y) \in \partial\Omega_i, i \in \{1,\dots,4\}, \end{cases}$$

kde u označuje teplotu uvnitř desky a f_i označuje teplotu na hranách desky.

Metoda 2D sítí

Pro řešení této úlohy využijeme metodu sítí ve 2D (více podrobností lze nalézt v [4]). Tato metoda je vhodná pro popis vedení tepla v rovině. Tato metoda poskytuje pouze přibližné numerické řešení a je silně omezena tvarem oblasti, pro čtvercovou desku je však plně postačující.

Výše uvedena Laplaceova rovnice matematicky popisuje model rozložení teploty, který odpovídá tomu, že teplota uvnitř desky se ustálí do rovnovážného stavu. Laplaceovou rovnici v našem případě vyřešíme pouze přibližně aproximací parciálních derivací u podle x a podle y:

$$\frac{\partial^2}{\partial x^2} u(x,y) \approx \frac{u(x-h,y) - 2u(x,y) + u(x+h,y)}{h^2}$$

$$\frac{\partial^2}{\partial y^2} u(x,y) \approx \frac{u(x,y-h) - 2u(x,y) + u(x,y+h)}{h^2}.$$

Použitím této aproximace dostaneme rovnici rovnováhy v libovolném uzlu:

$$-u_{i-1,j} - u_{i,j-1} + 4u_{i,j} - u_{i+1,j} - u_{i,j+1} = h^2 f_{i,j} = 0.$$

K rovnici rovnováhy se lze dostat i následující úvahou (viz [3]). Pro řešení úlohy rozložení teploty uvnitř desky budeme aproximovat teplotu pouze v konečném počtu bodů uvnitř desky. Tato aproximace je založena na principu střední hodnoty:

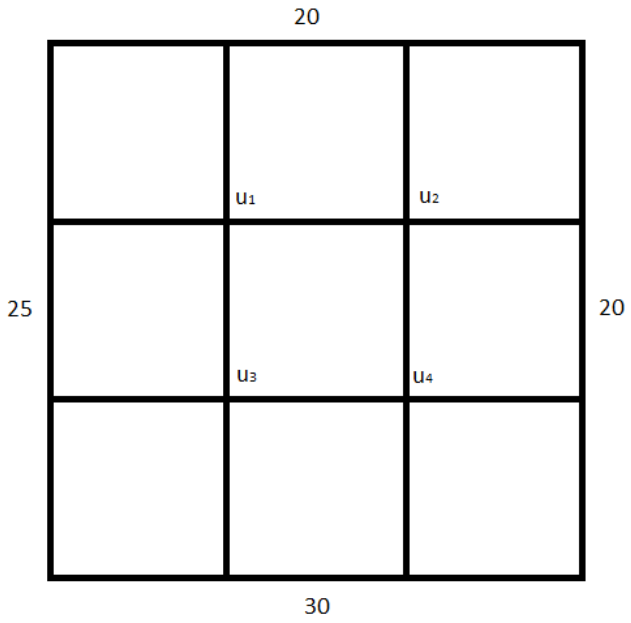
Pokud deska dosáhla tepelné rovnováhy a bod P je bodem uvnitř desky, C je kruh se středem v bodě P a je celý obsažen v desce, potom můžeme předpokládat, že v bodě P je dosaženo průměrné hodnoty tepelné funkce nad C.

Chceme-li zjistit, jak tato vlastnost funguje, umístíme na desku mřížku a vezmeme v úvahu body desky, se kterými se „setkávají" body mřížky. Nás budou zajímat teploty v těchto bodech pouze uvnitř desky. Navrhneme mřížku tak, aby některé z uvažovaných bodů ležely na hranici desky. Pro studium teploty v těchto bodech mřížky využijeme vlastnosti střední hodnoty popsané tvrzením:

Jestliže deska dosáhla tepelné rovnováhy a P je bod mřížky uvnitř desky, ale mimo její hranici, pak teplota v P je průměr teplot čtyř nejbližších bodů mřížky k P.

Příklad:

Začněme s mřížkou se čtyřmi vnitřními body a označme u_1, u_2, u_3 a u_4 jako teploty v těchto bodech. Situace je znázorněna na následujícím obrázku:



Po dosazení do rovnice rovnováhy v libovolném uzlu (rozměr desky neuvažujeme a položíme $h = 1$) dostaneme následující systém lineárních rovnic:

$$u_1 = \frac{20 + 25 + u_2 + u_3}{4}$$

$$u_2 = \frac{20 + 20 + u_1 + u_4}{4}$$

$$u_3 = \frac{25 + 30 + u_1 + u_4}{4}$$

$$u_4 = \frac{20 + 30 + u_2 + u_3}{4}$$

Po zjednodušení dostaneme:

$$4u_1 - u_2 - u_3 = 45$$
$$-u_1 + 4u_2 - u_4 = 40$$
$$-u_1 + 4u_3 - u_4 = 55$$
$$-u_2 - u_3 + 4u_4 = 50$$

Maticová forma systému je $AU = f$:

$$A = \begin{pmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{pmatrix}, U = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix}, f = \begin{pmatrix} 45 \\ 40 \\ 55 \\ 50 \end{pmatrix}$$

U nazýváme vektorem rovnovážných teplot. Řešení pro výše uvedeny systém je pak:

$$U = A^{-1} \cdot f$$

za předpokladu, že matice A je regulární.

Použitím metody pro výpočet inverzní matice, vypočteme A^{-1} a dostaneme:

$$A^{-1} = \begin{pmatrix} \frac{7}{24} & \frac{1}{12} & \frac{1}{12} & \frac{1}{24} \\ \frac{1}{12} & \frac{7}{24} & \frac{1}{24} & \frac{1}{12} \\ \frac{1}{12} & \frac{1}{24} & \frac{7}{24} & \frac{1}{12} \\ \frac{1}{24} & \frac{1}{12} & \frac{1}{12} & \frac{7}{24} \end{pmatrix}$$

Následně můžeme vypočítat vektor rovnovážných teplot z rovnice $U = A^{-1} \cdot f$ a dostaneme:

$$U = A^{-1} \cdot f = \begin{pmatrix} 23.125 \\ 21.875 \\ 25.625 \\ 24.375 \end{pmatrix}$$

Tento výsledek přímo znázorňuje teploty v bodech u_1, u_2, u_3 a u_4.

Příklad:

Teď předpokládejme, že teploty na okrajích desky se změnily z 25, 20 a 30 na 15, 10 a 20. Dostaneme potom nový systém lineárních rovnic, kde na levé straně zůstane stejná matice, ale vektor f se změní na:

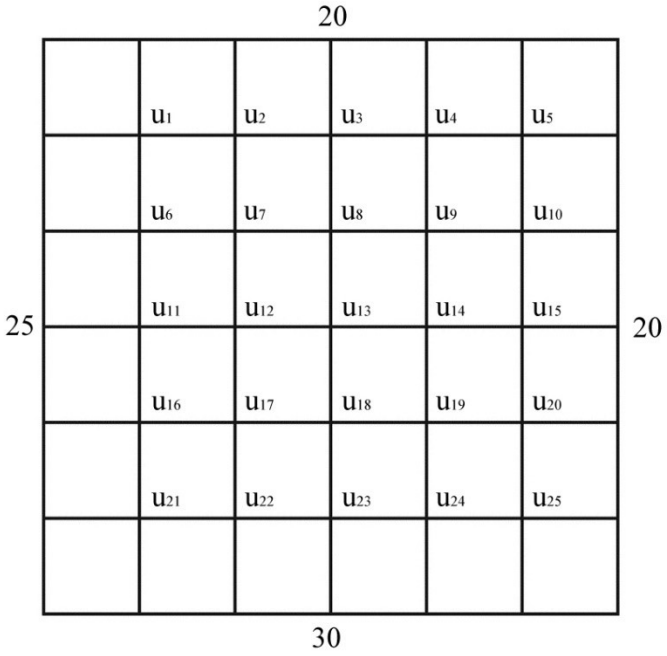
$$f = \begin{pmatrix} 35 \\ 20 \\ 35 \\ 30 \end{pmatrix}$$

Vektor rovnovážných teplot se nám v tomto případě změní na:

$$U = A^{-1} \cdot \begin{pmatrix} 35 \\ 20 \\ 35 \\ 30 \end{pmatrix} = \begin{pmatrix} 16.04 \\ 12.70 \\ 16.45 \\ 14.79 \end{pmatrix}$$

Příklad:

Výše uvedené aproximace teplot můžeme vypočítat mnohem přesněji, pokud použijeme jemnější mřížku, což znamená více vnitřních bodů pro výpočet. Podívejme se na desku z prvního příkladu, pokud mřížku zjemníme v každém směru:



Naše nová mřížka má teď 25 vnitřních bodů. Zopakováním stejného procesu použitého v předchozím příkladu, dostaneme systém 25 lineárních rovnic s 25 neznámými.

Matice soustavy pro úlohu vedení tepla A v tomto případě bude vypadat takhle:

$$\begin{pmatrix}
4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 4
\end{pmatrix}$$

Vektor f na pravé straně bude vypadat takto:

$$f = \begin{pmatrix}
45 \\ 20 \\ 20 \\ 20 \\ 40 \\ 25 \\ 0 \\ 0 \\ 0 \\ 20 \\ 25 \\ 0 \\ 0 \\ 0 \\ 20 \\ 25 \\ 0 \\ 0 \\ 0 \\ 20 \\ 55 \\ 30 \\ 30 \\ 30 \\ 50
\end{pmatrix}$$

Vyřešení této rovnice ručně by trvalo velmi dlouhou dobu, ale díky algebraickým programům, jako třeba Matlab, můžeme vypočítat inverzní matici velikosti 25 x 25 a následně vynásobit $A^{-1} \cdot f$ poměrně rychle. Výsledkem bude opět vektor U, který nám udává výsledné teploty v mřížce:

$$U = \begin{pmatrix} 22.65 \\ 21.76 \\ 21.28 \\ 20.89 \\ 20.46 \\ 23.86 \\ 23.10 \\ 22.49 \\ 21.81 \\ 20.98 \\ 24.69 \\ 24.30 \\ 23.76 \\ 22.90 \\ 21.65 \\ 25.60 \\ 25.68 \\ 25.29 \\ 24.39 \\ 22.72 \\ 27.03 \\ 27.51 \\ 27.36 \\ 26.65 \\ 24.84 \end{pmatrix}$$

V případě úlohy s větším počtem neznámých teplot je efektivnější postup pro nalezení řešení použití Gaussovy eliminační metody.

6 Vytvořené programy

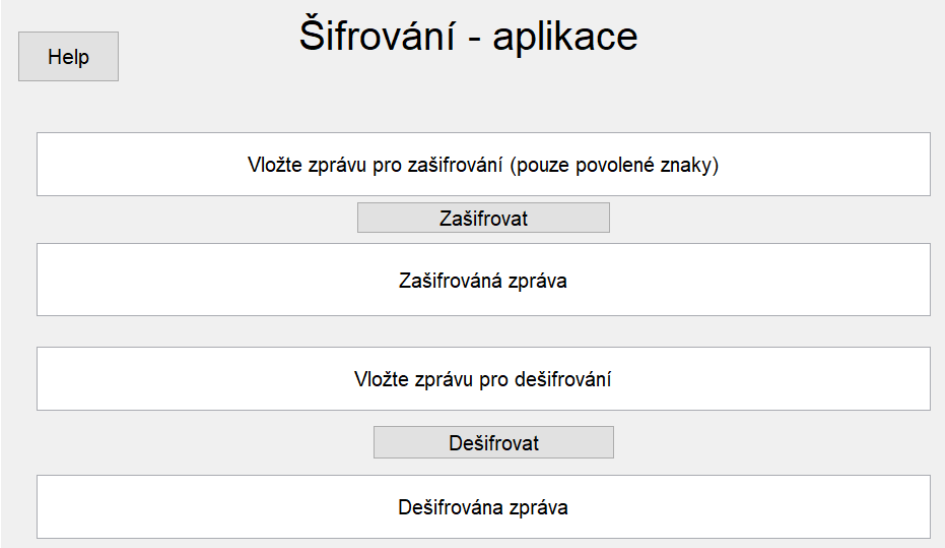
V této kapitole si představíme programy, které jsme vytvořili pro řešení příkladů s využitím lineární algebry. Pro vytvoření těchto programů jsme použili software Matlab a jeho nadstavbu GUIDE (Graphical User Interface Development Environment), kterou jsme využili pro vytvoření interaktivního prostředí a pro přidání grafické hodnoty.

Program Matlab má bohužel problémy se zobrazováním českých znaků a proto jsme v některých situacích byli nuceni použit znaky bez diakritiky.

6.1 Šifrování

První program, který jsme vytvořili, je program na šifrování textu. Tento program šifruje text pomocí způsobu popsaného v kapitole 2.

Po spuštění grafického prostředí příkazem „myEncoder“ se otevře okno, ve kterém máme čtyři řádky textu a dvě tlačítka.



Obrázek 1: Šifrovací program

První řádek slouží pro napsaní textu, který si přejeme zašifrovat. Jakmile máme text vložený, stiskneme tlačítko „Zašifrovat“ a v prostoru, kde je původně nápis „Vložte zprávu pro zašifrování (pouze povolené znaky)“ se nám zobrazí zašifrovaný řetězec čísel.

Obdobně postupujeme při dešifrování zprávy. Řetězec čísel, který chceme dešifrovat, jednoduše vložíme do řádku „Vložte zprávu pro dešifrování“ a stiskneme „Dešifrovat“. Dešifrovaný text se nám zobrazí ve čtvrtém řádku s nápisem „Dešifrována zpráva“.

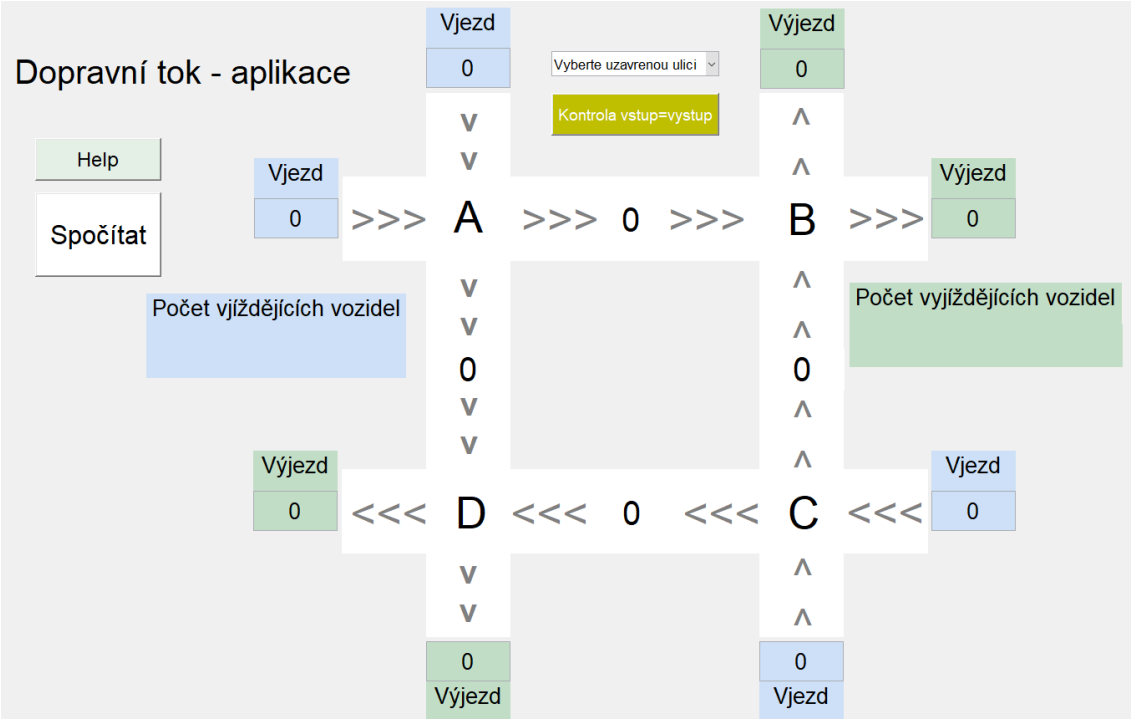
Ukažme si to na příkladu z druhé kapitoly „ahoj karle“.

Obrázek 2: Ukázka výsledku šifrovacího programu

6.2 Dopravní tok

Další vytvořený program nám řeší problém s dopravním tokem a plynulosti dopravy v odpolední špičce.

Spuštěním programu příkazem „TrafficGUI“ se nám objeví následující okno:



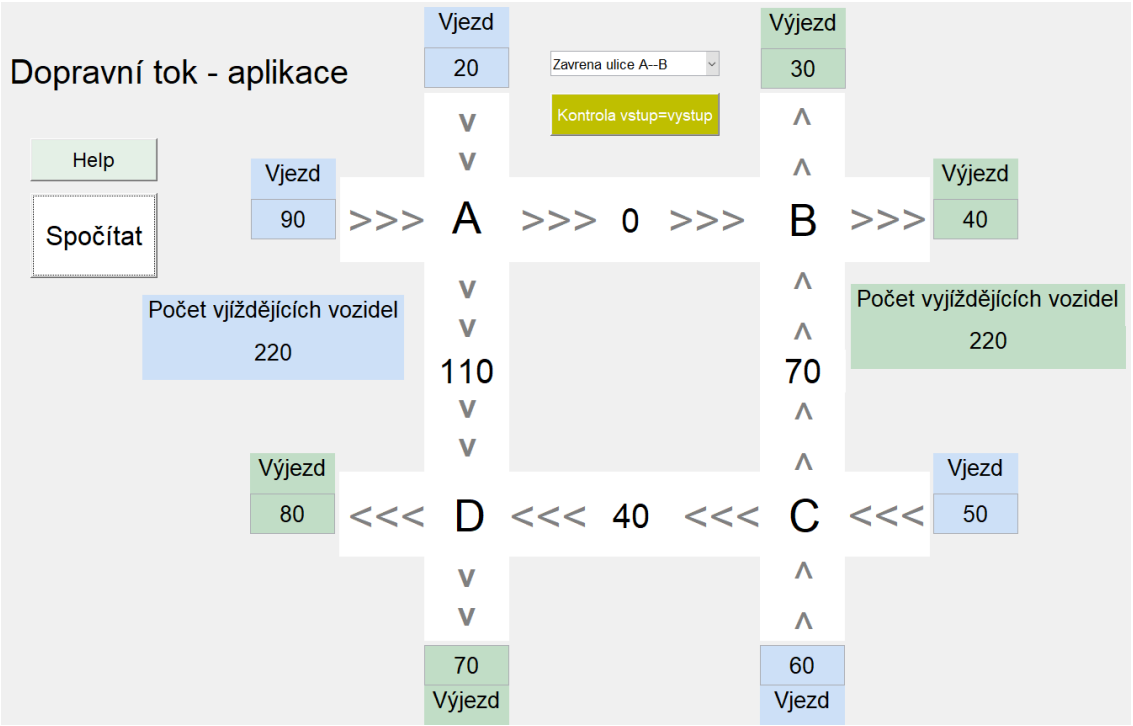
Obrázek 3: Dopravní tok

Umístění příslušných počtu vozů, které přijíždějí do křižovatek jsou znázorněny přímo v programu. Počet vozidel přijíždějící do křižovatek je označen modrou barvou a počet vozidel vyjíždějících z křižovatek je označen barvou zelenou.

V programu můžeme rovnou zkontrolovat, jestli počet vozidel přijíždějících do křižovatek se rovná počtu vozidel vyjíždějících stisknutím tlačítka „Kontrola vstup=výstup“.

Ve třetí kapitole jsme si ukázali, proč nemá smysl řešit tuto situaci, když jsou všechny ulice otevřené, a proto v horní části programu máme možnost si vybrat uzavření jedné z ulic.

Jakmile máme všechny naměřené hodnoty vložené do příslušných okének a vybranou uzavřenou ulici, stiskneme tlačítko „Spočítat“ a program nám zobrazí, kolik vozidel pojede kterým směrem.

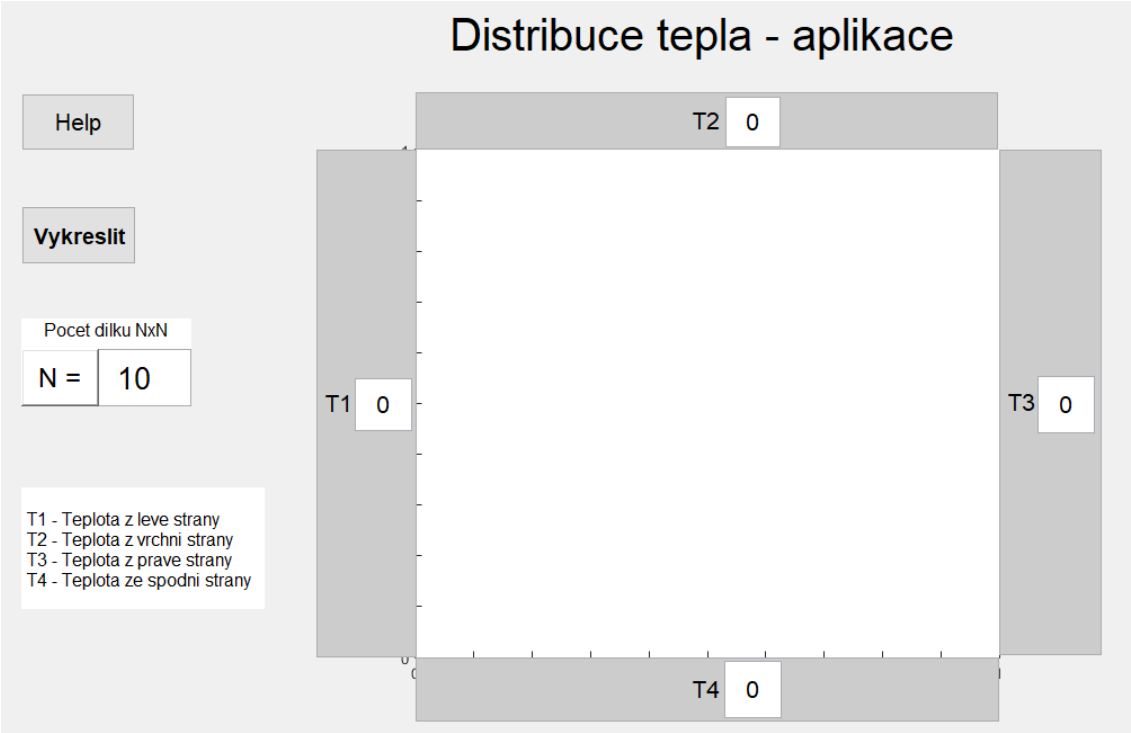


Obrázek 4: Ukázka výsledku programu na dopravní tok.

6.3 Distribuce teploty

Posledním vytvořeným programem je program, který nám znázorní rozložení tepla ve čtvercové desce pomocí vytvoření mřížky na desce a následné aproximaci teplot v jednotlivých bodech.

Po spuštění programu příkazem „Temperature“ dostaneme následující okno:

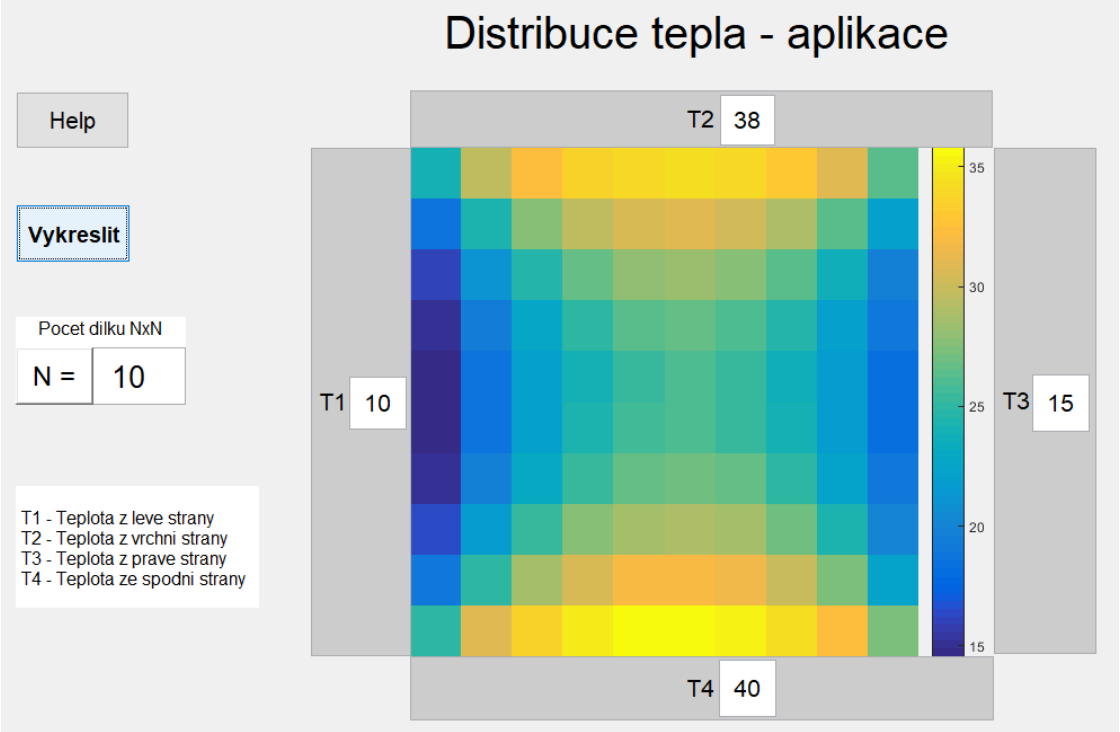


Obrázek 5: Program pro distribuci teploty.

V programu se nám zobrazí čtyři hodnoty pro vyplnění a čtvercová deska uprostřed pro vykreslení teplot v jednotlivých bodech.

V levém horním rohu si můžeme vybrat, jak bude výsledný graf jemný. Jako výchozí hodnota je nastaveno 10 dílků, což znamená, že mezi levou a pravou stranou desky bude 10 různých hodnot teploty.

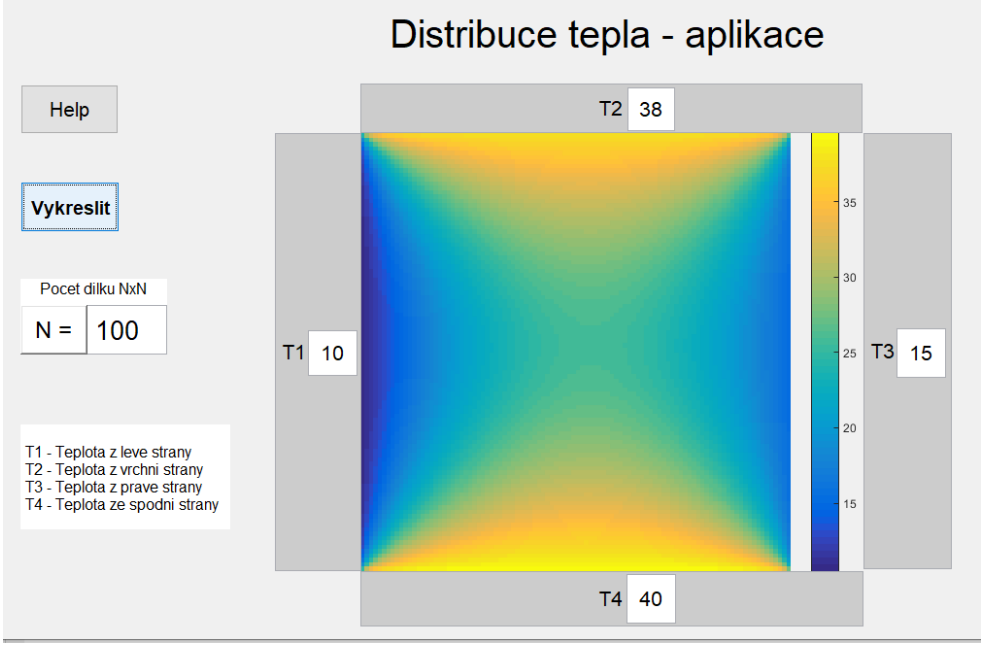
V první ukázce ponecháme počet dílků 10, vybereme teploty na krajích desky a stiskneme „Vykreslit“.



Obrázek 6: Jemnost mřížky – 10 dílků v každém řádku a sloupci.

Z tohoto obrázku si můžeme udělat určitou představu o rozložení teploty uvnitř desky, nicméně tento výsledek je hodně nepřesný.

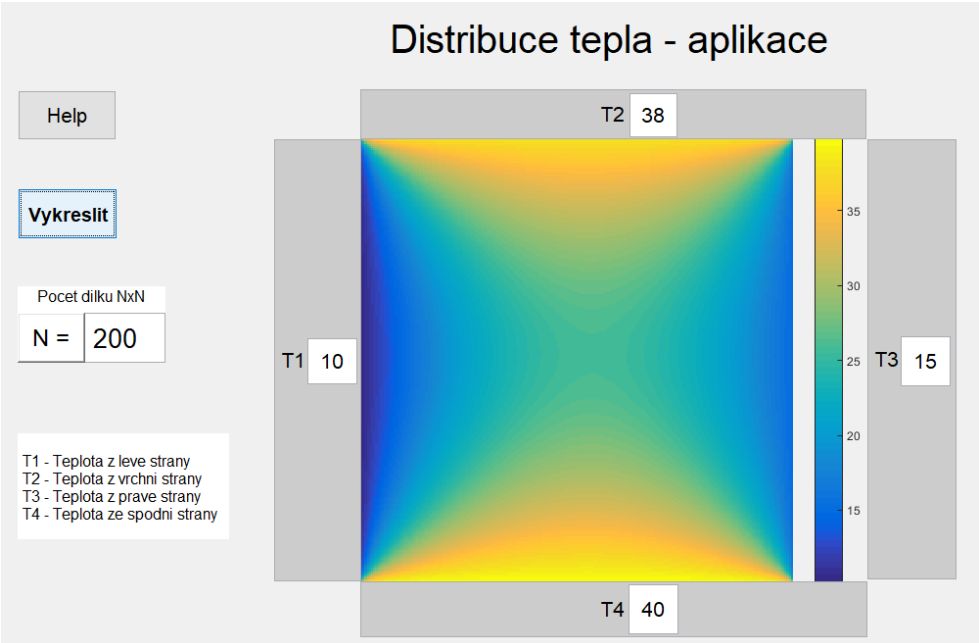
Povídejme se na výsledek, pokud jemnost mřížky zvýšíme mnohonásobně na 100.



Obrázek 7: Jemnost mřížky – 100 dílků v každém řádku a sloupci.

Vykreslený graf se mnohonásobně zjemnil a už si můžeme udělat docela přesnou představu a rozložení tepla v takové desce při vybraných teplotách.

Pojďme se ale podívat, co se stane, když mřížku zjemníme na 200 dílků.



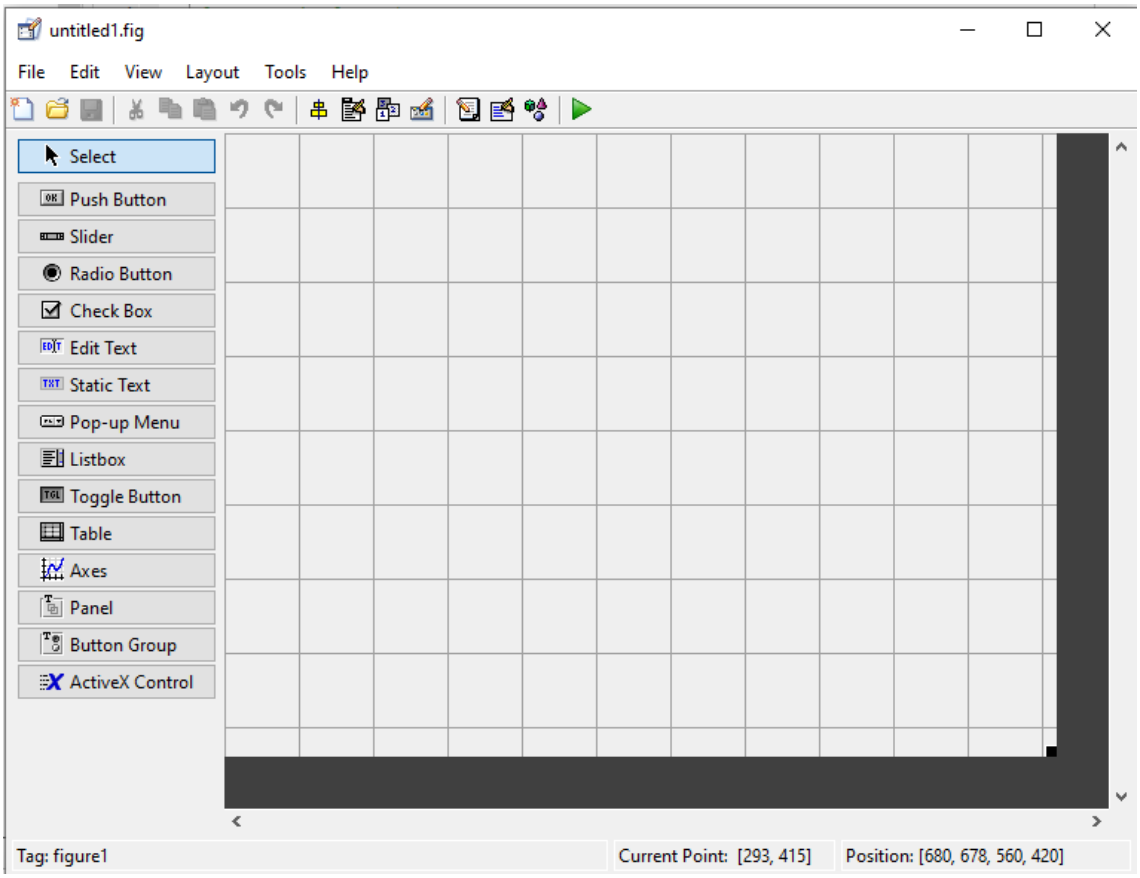
Obrázek 8: Jemnost mřížky – 200 dílků v každém řádku a sloupci.

Ihned si všimneme, že rozdíl v jemnosti obrázku při malém rozlišení je nepatrný, ale rozdíl v době, který program potřeboval k vyřešení se mnohonásobně prodloužil.

6.4 Implementace

Všechny vytvořené aplikace byly vytvořeny v grafickém prostředí Matlabu – GUIDE (Graphic User Interface Development Environment). V této kapitole si znázorníme jejich implementaci.

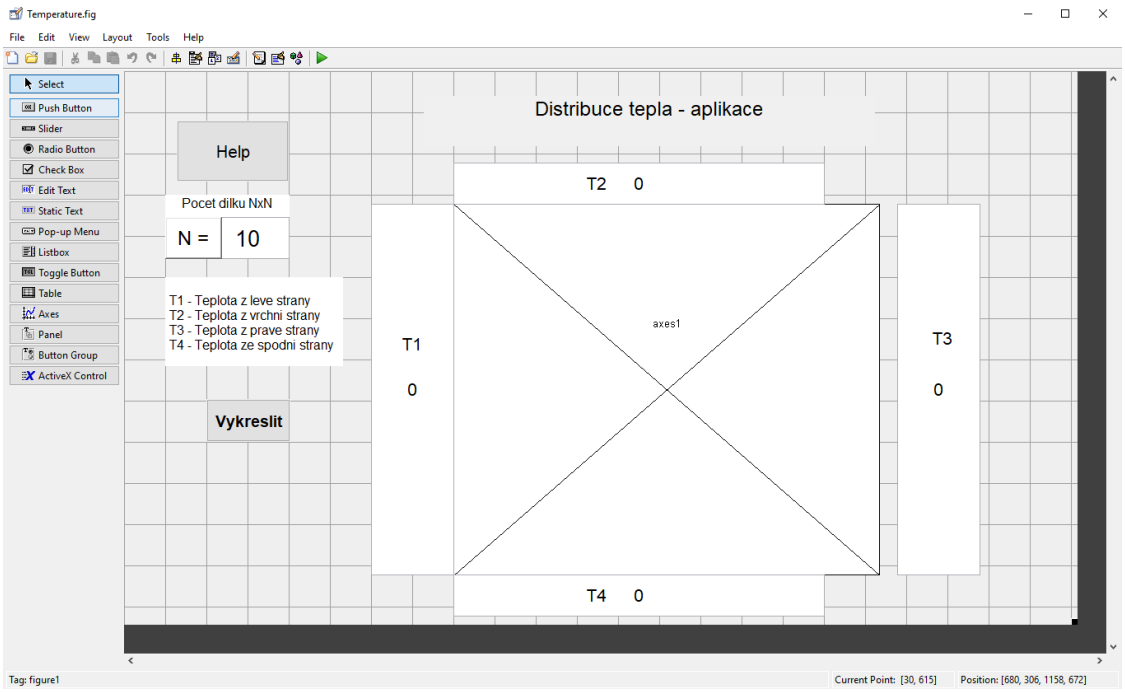
Vytvoření grafického prostředí spustíme příkazem „guide“ v příkazovém řádku Matlabu, vytvoříme si nový GUI (Graphic User Interface) a dostaneme následující okno.



Obrázek 9: Uvítací obrazovka prostředí GUIDE.

Toto okno obsahuje ovládací prvky na levé straně, které můžeme libovolně rozmístit. Pomocí callback funkcí pak ovládáme jednotlivé komponenty a řídíme běh programu. V tomto kroce rozhodujeme o tom, jak bude program vypadat a co všechno bude obsahovat.

Vezměme si jako příklad poslední aplikaci na distribuci tepla. Nejdříve si vytvoříme potřebné prvky pro naši aplikaci:



Obrázek 10: Prvky prostředí GUIDE.

Pro tuto aplikaci jsme využili následující prvky:

Push Button – pro vytvoření tlačítek Help a Vykreslit

Static Text – pro nadpis Distribuce tepla – aplikace, Počet dílků NxN a podobně

Edit Text – pro zvolení počtu dílků a vložení teplot

Axes – pro vykreslení výsledného grafu.

Když máme hotové rozvržení GUI, vytvoří se nám 2 soubory, jeden .fig, kde jsou uloženy data z GUI a druhý .m, do kterého můžeme vkládat a volat funkce.

Například, po spuštění tlačítka Vykreslit se zavolá funkce calculate1, ve které máme celý kód pro distribuci teploty. Začátek kódu vypadá následovně:

```
function calculate1_Callback(hObject, eventdata, handles)

a1 = get(handles.temp1,'String');
a2 = get(handles.temp2,'String');
a3 = get(handles.temp3,'String');
a4 = get(handles.temp4,'String');
n_temp = get(handles.n_editText,'String');

t1 = str2num(a1);
t2 = str2num(a2);
t3 = str2num(a3);
t4 = str2num(a4);
n = str2num(n_temp);
```

Proměnnými a1-a4 dostáváme hodnoty vložené v aplikaci jako okrajové teploty desky. Hodnota n je počet dílků. Tyto hodnoty jsou ve formátu string a proto je následně proměnnými t1-t4 změníme na formát number, abychom s nimi mohli pracovat.

Následně si vytvoříme řídkou matici soustavy pro úlohu vedení tepla A:

```
A = sparse(n*n,n*n);
i = 1;
grid = n*n;

while i <= grid
    A(i,i) = 1;
    i = i + 1;
end

i = 1;
while i <= ((n*n)-n)
    A(i,i+n) = -0.25;
    A(i+n,i) = -0.25;
    i = i + 1;
end

i = 1;
while i <= ((n*n)-1)
    A(i,i+1) = -0.25;
    A(i+1,i) = -0.25;
    if mod(i,n) == 0;
        A(i,i+1) = 0;
        A(i+1,i) = 0;
    end
    i = i + 1;
end
A;

b=1:n;
b=b';
i = 1;
while i<=n
    b(i) = 0;
    i = i + 1;
end
```

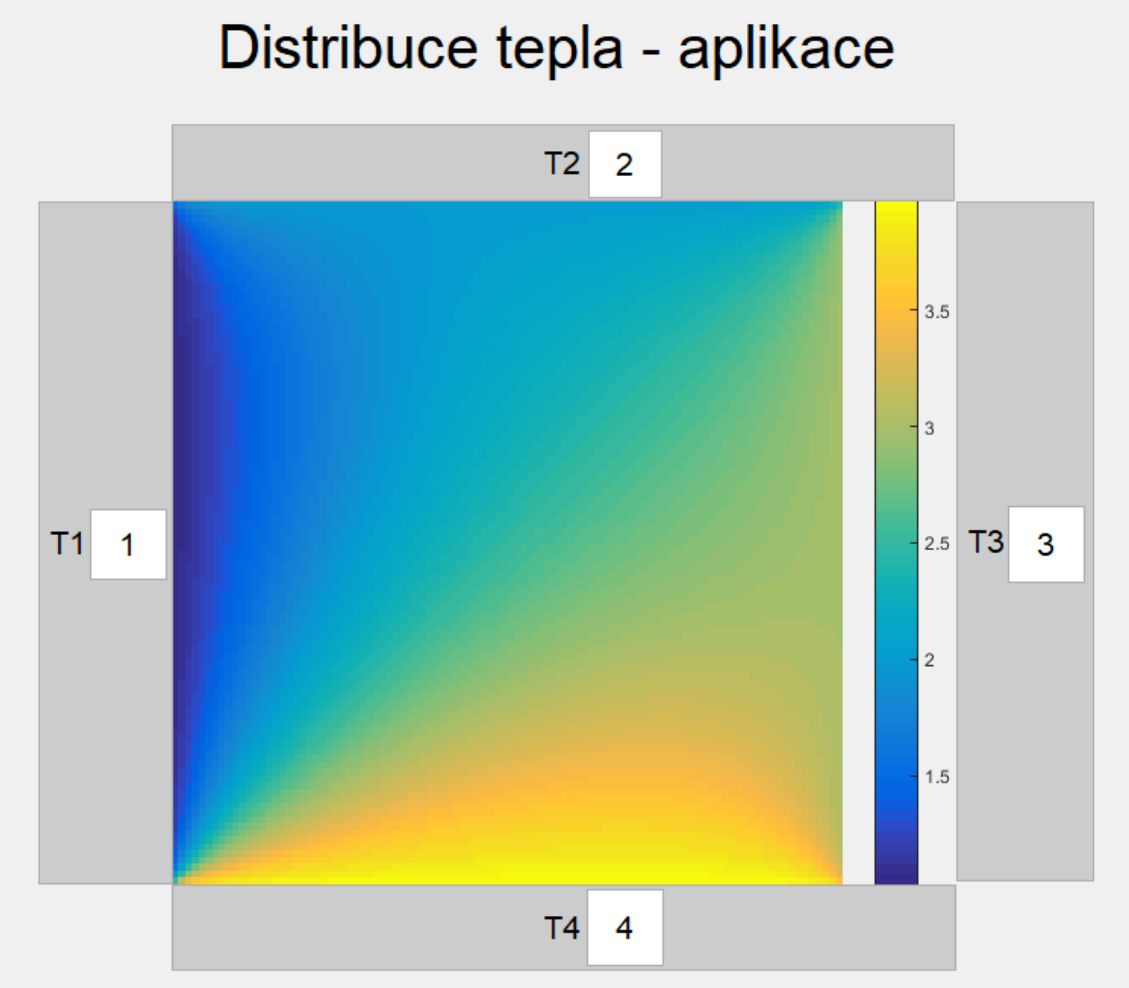
Dále si vytvoříme vektor pravé strany b ze zadaných hodnot teploty na okrajích desky a vyřešíme rovnici $A \cdot X = b$ pomocí Matlabového příkazu pro řešení soustavy lineárních rovnic pomocí Gaussovy eliminační metody:

$$X = A \backslash b$$

Hodnotu si uložíme do proměnné a vykreslíme do prostředí GUI do prvku axes,

```
axes(handles.axes1);
imagesc(B);
colorbar;
axis off;
```

kde se nám zobrazí výsledný graf:



Obrázek 11: Výsledný graf

7 Závěr

V bakalářské práci jsme se seznámili se základními pojmy lineární algebry. Vybrané pojmy a operace jsme si podrobněji popsali a uvedli příslušné příklady. Popsali jsme si různé druhy matic, seznámili jsme se s postupem pro získání inverzní matice a s výpočtem soustavy lineárních rovnic pomocí Gaussovy eliminační metody. Seznámili jsme se taky s pojmem lineární zobrazení a uvedli jsme si jeho vztah s aplikací pro šifrování. Následně jsme vytvořili 3 aplikace, ve kterých využíváme postupy popsané v předchozích kapitolách. Tyto aplikace – šifrování, dopravní tok a rozložení teploty mohou být použity ve výuce lineární algebry.

Literatura

[1] BEREMLIJSKI,P., *Lineární zobrazení* [online]:prezentace k přednášce. VŠB – Technická Univerzita Ostrava, 2012. [cit. 2019-07-10]
Dostupné z: https://homel.vsb.cz/~ber95/LA/Prednasky/LA_7.pdf>

[2] DOSTÁL, Z., VONDRÁK, V. *Lineární algebra* [online]:skriptum. VŠB – Technická Univerzita Ostrava, 2012. [cit. 2019-07-10]
Dostupné z: http://mi21.vsb.cz/sites/mi21.vsb.cz/files/unit/linearni_algebra.pdf

[3] HUSSEIN, M., *Application of Systems of Linear Equations* [kniha]. United Arab Emirates University – College of Engineering & College of Science, 2009. [cit. 2019-07-10]

[4] KOZUBEK, T., BRZOBOHATÝ, T., HAPLA, V., JAROŠOVÁ, M., Alexandros, M., *Lineární algebra s Matlabem* [online]:skriptum. VŠB – Technická Univerzita Ostrava, 2012. [citováno 10.7.2019]
Dostupné z: http://mi21.vsb.cz/sites/mi21.vsb.cz/files/unit/skripta_lam_obr.pdf

[5] *Wikipedie: Otevřená encyklopedie: Historie kryptografie* [online]. Datum poslední revize 19.5.2019, [citováno 2019-07-10]
Dostupné z: https://cs.wikipedia.org/wiki/Historie_kryptografie

[6] *Wikipedie: Otevřená encyklopedie: Enigma* [online]. Datum poslední revize 19.6.2019, [citováno 2019-07-10]
Dostupné z: https://cs.wikipedia.org/wiki/Enigma